A COMPARISON OF MC68000 FAMILY PROCESSORS

BY THOMAS L. JOHNSON

A look at the architecture and hardware and software compatibility

SINCE ITS 1979 introduction, the Motorola MC68000 microprocessor family has grown to five members: the original 16-bit MC68000; the MC68008, an 8-bit data bus version of the MC68000; the MC68010, a virtual memory/virtual machine version; the MC68012, an extended address version of the MC68010; and the MC68020, a full 32-bit virtual memory/virtual machine microprocessor (see figure 1).

Each family member includes architectural enhancements that offer certain advantages, in terms of software, hardware, and system design. However, the extent of software and hardware compatibility between the family members varies.

FAMILY OVERVIEW

The major external differences are in the number of address, data, and various control lines (see table 1).

The MC68000 family uses a three-wire function code that transmits to the system the processing state of the processor on a bus-cycle-by-bus-cycle basis. This allows the designer to maintain in the system hardware the memory privilege distinctions maintained within the processor. For the MC68000 and MC68008 these function codes can indicate either supervisor mode or user mode, program accesses or data accesses, and interrupt acknowledge bus cycles. The MC68010 and MC68012 use the same encodings on the function code pins but change the name of the interrupt acknowledge space to "CPU space." A CPU space access indicates either interrupt acknowledge or breakpoint acknowledge. The MC68020 expands CPU space accesses to include coprocessor communications and access-permission-level checking, which can be used in sophisticated memory management schemes to implement ring protection architectures. All encodings are done in an upwardly compatible manner, such that systems that recognize an interrupt acknowledge on an MC68000 will work properly even when an MC68020 is inserted in the system. (See figure 2.)

USER PROGRAMMING MODEL

To the user programmer, there is absolutely no difference in the on-chip resources available for any of the processors in the family. Thus, once a model has been assimilated, no retraining is needed when changing from one family member to any other. The base architecture for the user incorporates eight totally undedicated 32-bit data registers, which can be accessed as bytes, words, or long words (32-bit values) and used for either the source or destination of any operation that will allow the use of a data register (i.e., all arithmetic, logical, and data movement operations). Users also have at their command eight 32-bit address registers that can be accessed as either 16-bit or 32-bit entities. Only one of these registers may be considered dedicated—A7 is the implicit user stack pointer for subroutine calls and so forth.

Rounding out the user model is an 8-bit condition code register and a full 32-bit program counter. The condition code register contains not only the "normal" condition bits for arithmetic

(continued)

Thomas L. Johnson is a staff engineer at Motorola Inc. (6501 William Cannon West, Austin, TX 78735-8598).

and logical operations (zero, carry, overflow, negative), but also a bit for extended-precision operations (extended).

The inclusion of 32-bit structures foreshadowed the design of the full 32-bit MC68020—setting the stage for upward compatibility of user-level code in the future processors.

SUPERVISORY PROGRAMMING MODEL

A look at the supervisory-level programming model (figure 3) finally reveals the differences in functionality of the family members. Whereas the user-level program has access only to user-level resources, the supervisory-level program has access not only to the complete user-level model, but also to the full supervisory-level programming model for that processor. Note that the supervisor stack pointers are also referred to as A7. An automatic context switch concerning these stack pointers occurs as flow changes from supervisory to user level and back again without the necessity of loading and reloading stack pointer registers. Table 2 encapsulates all of the resources available to the supervisory-level program on the various processors.

Progressing from the MC68000 to the MC68020, more instructions are available to manipulate the additional supervisory resources, and each model is a proper subset of the next higher processor. Thus, code that manipulates the condition code register on the MC68008 (the 8-bit version of the MC68000) will execute intact on any other of the processors.

This form of "supersetting" the resources on newer family members ensures upward compatibility and is maintained within the family for the supervisor and user programming models, the instructions (to the bit level), the address modes, and all data types.

ADDITIONAL SUPERVISORY-LEVEL RESOURCES

The vector base register (VBR), introduced on the MC68010, allows a supervisory-level program to relocate the 1K-byte exception vector table to anywhere in the physical address map of the processor. This exception vector table is used in the MC68000 family to route specific exceptions to the appropriate handler within the supervisory code space. The vector table consists of 256 four-byte entries, each of which may be used to point to the entry address of its specific handler. Of these 256 vector entries, 37 to 39 vectors (depending on the processor) are dedicated to handling predefined exceptions such as bus errors, TRAP instructions, and so forth. An additional 26 vectors have been reserved for future expansion of functionality, and 192 vector entries can be defined by the user.

The alternate function code registers (SFC and DFC) allow the supervisor to retrieve and modify data in *any* address map (user or supervisor, program or data). On all family members, the function code outputs are driven automatically based on the

(continued)

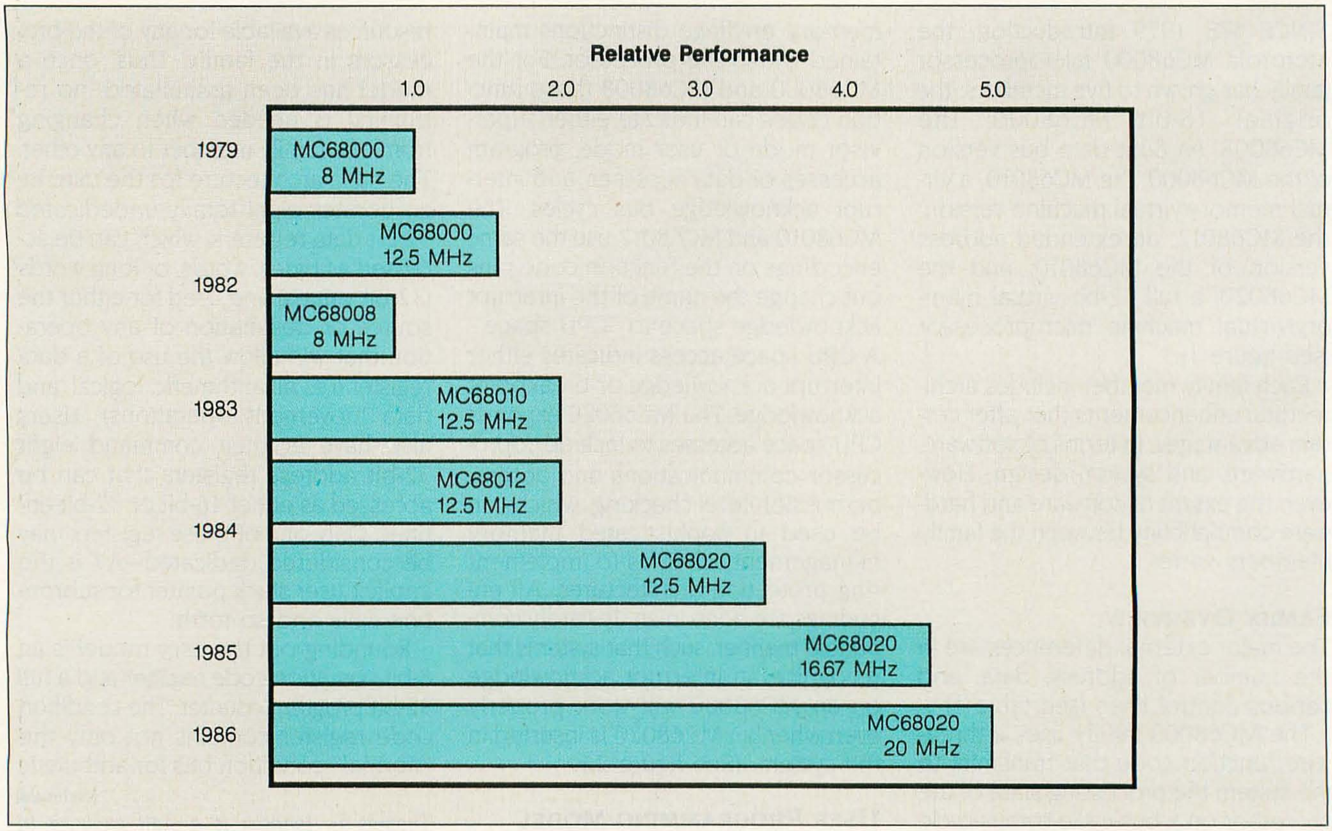


Figure 1: *The* MC68000 *family genealogy.*

type and location of the memory bus cycle, and these outputs may be used to enforce hardware protection in the memory subsystem. Because the MC68000 and MC68008 are not virtual machine processors, they have no need to override this hardware-enforced protection, but the MC68010, MC68012, and MC68020 processors require this ability. The alternate function code registers allow these newer processors to perform that function to control the virtual environment.

The two cache control registers introduced in the MC68020 (the cache control register—CACR, and the cache address register—CAAR) allow the supervisor to control the on-chip instruction cache memory (64 four-byte entries) for that device. The cache powers up as disabled, and the supervisor, at its discretion, may choose to enable it at some later time. Additionally, the supervisor can selectively invalidate cache entries.

The MC68020 status register contains an additional bit for control of the hardware tracing mechanism built into all the family members. The MC68000, MC68008, MC68010, and MC68012 can trace any given instruction or sequence of instructions when the supervisor enables the tracing mechanism. The MC68020 can also trace only on a program change of flow instruction, such as branch or jump.

The only other added supervisory resource is the master stack pointer (MSP), also new on the MC68020, which allows the separation of task-related and non-task-related exception stacking. In use, the MC68020 powers up and loads the interrupt stack pointer as the active supervisory stack pointer. This is the equivalent to the power-up sequence of any other family member in which the single supervisory stack pointer is loaded. At some later time, the operating system may choose to enable the master stack pointer by enabling the MSP bit in the status register (the stack pointer select bit). From that point on (until disabled), all common program exceptions, including TRAPs and divide-by-zero, that occur will cause state information to be saved on the master stack until an interrupt occurs.

Since interrupts are asynchronous events that will normally have no bearing on the task currently executing in a multitasking environment, it is desirable to separate the interrupt-related information from the task-related information. This is accomplished by first stacking the task-related information from the current task on the master stack and then performing an automatic switch to the interrupt stack pointer for use during interrupt handling away from the interrupted task. Here, a "dummy" stack frame is stored on the interrupt stack so that at the completion of interrupt processing, a single RTE (return from exception) instruction will automatically cause a switch back to the master stack pointer, for reloading the previously saved task state and resumption of that task. The benefit is that the supervisor need not provide stack space for interrupts that may occur during normal task execution and may be unrelated to the currently executing task.

INSTRUCTION SETS

On the MC68020, the RTE instruction can perform a large number of dis-

(continued)

Table 1: *A pin-out comparison.*

	MC68000	MC68008	MC68010	MC68012	MC68020
Address lines	24	20/22	24	31	32
Virtual address range	16Mb	1Mb/4Mb	16Mb	2Gb	4Gb
Data lines	16	8	16	16	32
Bus protocol	async	async	async	async	async
Clock cycles/bus cycle	4	4	4	4	3
Number of data strobes	2	1	2	2	1
Number of transfer acknowledges	1 (DTACK)	1 (DTACK)	1 (DTACK)	1 (DTACK)	2 (DTACK)
Function code lines	3	3	3	3	3
Bus arbitration lines	3	2/3	3	3	3

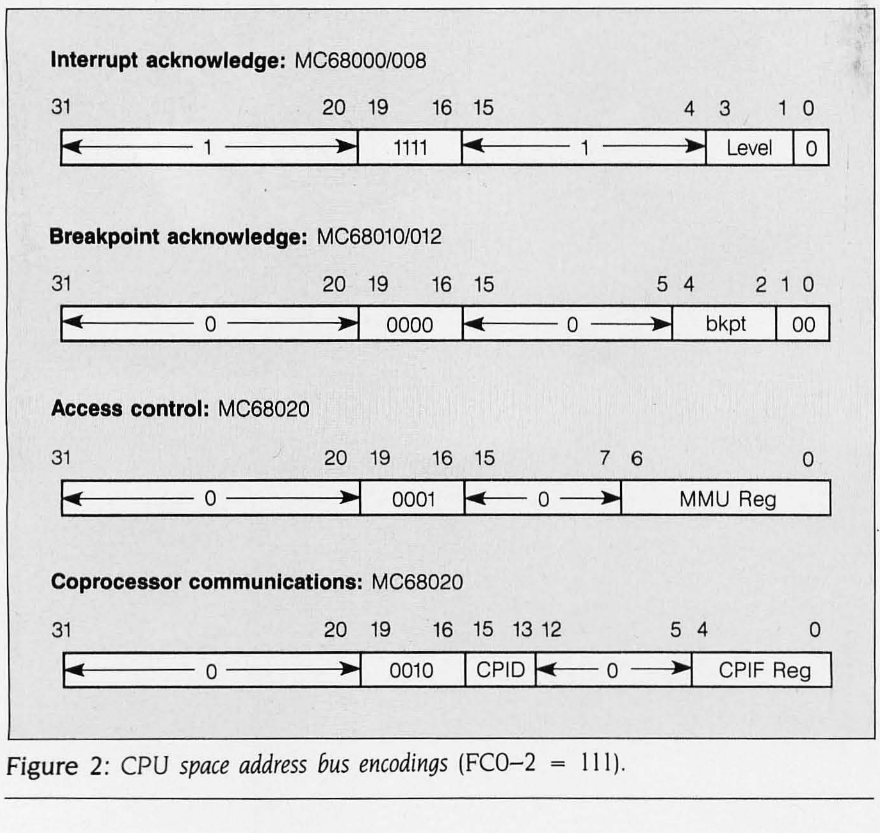


Figure 2: *CPU space address bus encodings* (FC0–2 = 111).

crete functions (check for dummy frame, change stack pointers, reload state); however, this is the same instruction used on the MC68000, MC68008, MC68010, and MC68012. The function performed by the RTE will relate to the resources and functionality available on the respective processors. Thus, for the MC68000 and MC68008 to return from interrupt processing or normal exception handling, this instruction simply restores the status register and program counter of the suspended task from the supervisory stack pointer, whereas on the MC68010 and MC68012 it also checks the stack format and can reload the more substantial amount of information for the saved internal state. On all processors, the RTE instruction performs whatever is necessary to return from an exception.

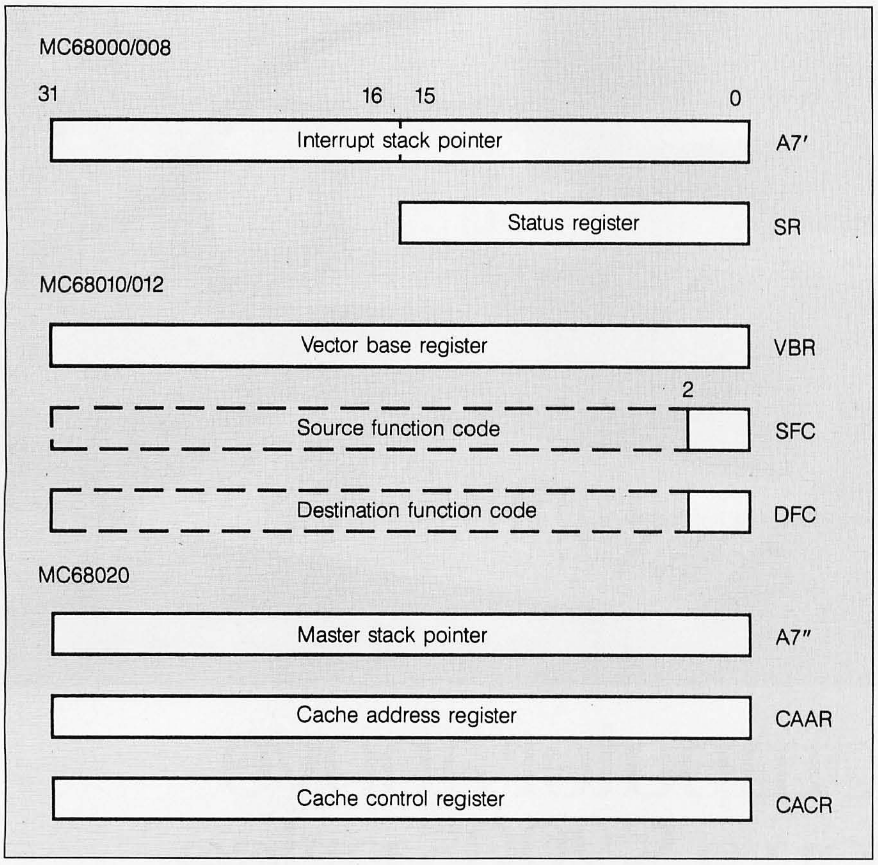


Figure 3: *The supervisory programming model.*

Table 2: A *comparison of supervisory resources.*

	MC68000/008	MC68010/012	MC68020
Status register			
Hardware trace bits	1	1	2
Supervisor state bit	yes	yes	yes
Interrupt mask bits	3	3	3
Stack pointer select	no	no	yes
Stack pointers			
Interrupt stack	yes	yes	yes
Master stack	no	no	yes
Vector base register	no	yes	yes
Cache control registers	no	no	yes
Alternate function codes	no	yes	yes

The instruction set of the base architecture, that of the MC68000, consists of a set of user-available instructions and a set of *privileged* instructions for control of sensitive system resources. The later family MPUs enhanced this instruction set (see table 3).

Address Modes

As with the instruction sets, the address modes are also upwardly compatible from one family member to another. All address modes on MC68000 family processors can be generally divided into four main categories: *data*, if an address mode may refer to data operands; *memory*, if the address mode may be used to refer to operands in memory; *control*, if the processor needs absolutely no knowledge of the size of the operand prior to calculation of the effective address; and *alterable*, if the mode may refer to writable operands (see table 4). Since it is possible to refer to an alterable data operand in memory, it follows that any given mode may be classified in more than one category.

The address modes in table 4 are exactly the same for the MC68000, MC68008, MC68010, MC68012, and MC68020 microprocessors. These modes provide a substantial amount of flexibility and power. For instance, the (An)+ and −(An) modes allow for the maintenance of up to eight stacks per task, a particularly attractive capability in the artificial intelligence arena. To these modes, the MC68020 adds one more mode: memory indirect. The memory indirect mode has a large number of variations and can best be illustrated by the syntactical expressions

([bd,An],Xn.size*scale,od) and

([bd,An,Xn.size*scale],od)

In these new modes, bd and od refer to optional sign-extended 8-, 16-, or 32-bit base and outer displacements; An, to an optional base address register (A0–A7); Xn, to an optional index register (D0–D7 or A0–A7); size, how much of the index register is to be used (16 or 32 bits); and scale, to a scaling factor for the index register (1, 2, 4, or 8).

(continued)

Because the memory indirect mode allows any of the base displacement, base address register, index register, and outer displacement to be optional, and allows the index register to be added to the address calculation either before or after the first address access (pre- and post-indexing), any conceivable combination of these modes is allowed. This includes displaced memory indirect ([bd],od), data register indirect (Dn); and register doubly indirect ([Rn]).

The addition of these new modes, however, in no way impacts the compatibility of the earlier modes. Any address mode legal for the MC68000, MC68008, MC68010, and MC68012 operates exactly the same on the MC68020.

Virtual Operation

Due to the internal architecture of the processors, there are three basic machines: the MC68000/008 architecture, the MC68010/012 architecture, and the MC68020 architecture. In a nutshell, any user-level code written to execute on the MC68000/008 will execute unchanged on any of the other processors, and any user-level code written to execute on the MC68010/012 will execute unchanged on the MC68020. This upward compatibility applies even to the binary level. The major difference between the processors insofar as exception processing is concerned is in the state frame information that is stored as a result of an exception in support of virtual memory and virtual machine operation.

Virtual memory is a technique that allows all tasks executing on a processor to behave as if each had at its disposal the entire addressing range of the processor, regardless of the amount of physical memory present in the system. Virtual machine is a technique in which all processor and system resources required by a task appear to be present, even if not implemented. To support the virtual memory capability, any given instruction must be "abortable" and in some manner "restartable."

For example, suppose that task A is executing a 32-bit memory-to-memory MOVE instruction and the physical piece of memory required for the destination operand is not currently available. The fetch of both the instruction and the source operand are successfully accomplished, but when the write to the destination is attempted, the memory manager indicates a "not-resident" fault by means of a bus error or fault line to the processor. The processor must then save the state of task A, find an empty piece of memory (possibly by swapping a currently resident piece to a disk), load the target piece of memory from the disk, restore task A, and allow the MOVE instruction either to restart or complete. So, from the time of the fault until the completion of the MOVE instruction, task A was not executing but has no indication of this fact.

Although all MC68000 family processors have the "abort" capability

(continued)

Table 3: *A comparison of instruction sets.*

	MC68000/008	MC68010/012	MC68020
Data movement	7	9	9
Arithmetic/logical	17	17	18
Binary-coded decimal	5	5	7
Single operand	9	9	9
Shift and rotate	8	8	8
Bit manipulations	4	4	4
Bit-field manipulations	0	0	8
Branches (16 conditions)	3	3	3
Exception-related	5	7	8
Control	12	13	20
Coprocessor generic instructions	0	0	7
Total	70	75	101

Table 4: *The base architecture address modes.*

Mode	Syntax	Data	Memory	Control	Alterable
Data register direct	Dn	•			•
Address register direct	An				•
Address register indirect	(An)	•	•	•	•
Address register indirect with post-increment	(An)+	•	•		•
Address register indirect with pre-decrement	–(An)	•	•		•
Address register indirect with displacement	d(An)	•	•	•	•
Address register indirect with index	d(An,Rx)	•	•	•	•
Absolute short	xxx.W	•	•	•	•
Absolute long	xxx.L	•	•	•	•
PC relative with displacement	d(PC)	•	•	•	
PC relative with index	d(PC,Rx)	•	•	•	
Immediate	#xxx	•	•		

Dn refers to any of the eight 32-bit data registers, D0–D7.
An refers to any of the eight 32-bit address registers, A0–A7.
Rx refers to any 32-bit register, D0–D7 or A0–A7.
d refers to an 8- or 16-bit sign-extended displacement.
+, – refer to the automatic post-incrementing/pre-decrementing of the specified address register by 1, 2, or 4 depending on the size of the target operand (1, 2, or 4 bytes).

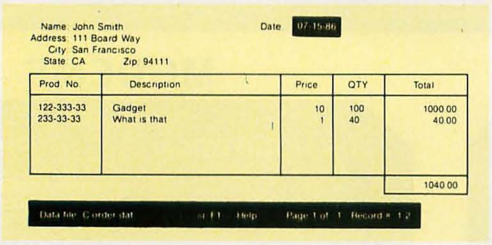
via the bus error (BERR) input of the processors, the MC68000 and MC68008 are not virtual memory processors and therefore do not have all the features necessary to support virtual memory. Specifically, insufficient internal processor state information is available on receipt of a bus fault to allow recovery of the faulted instruction. The other family members (MC68010, MC68012, and MC68020), however, save sufficient state information to allow complete restoration of the faulted instruction. The faulted instruction should be recovered by one of two methods: *instruction restart*, in which the processor is "backed up" to the start of the instruction that caused the fault, or *instruction continuation*, in which the faulted instruction is allowed to complete execution from the point of the fault. The choice of method may seem to be a matter of computer science dogma, but actually it is of practical importance. The advanced address modes on the MC68000 family and the desire to protect the user from "processor thrashing" caused the designers of the MC68000 family to choose the instruction continuation method.

INSTRUCTION CONTINUATION

Processor thrashing can be a very debilitating problem for virtual memory systems. The example used above can also be used to illustrate this problem. In the example, it was assumed that the op code and source operand fetches occurred without incident, but a fault was taken on the operand write. Since most memory replacement algorithms look for unmodified segments to be swapped out (because these segments don't need to be written to a disk prior to releasing the memory space), a good candidate can be the page containing the source operand. If this segment is chosen to be swapped with the destination page, then when the instruction is restarted (rather than continued), the instruction will once again fault—this time for the fetch of the source operand. Thrashing has now begun, with the source and destination areas continually being swapped with one another (since no writes to memory will ever occur), and the instruction will never complete. Some replacement algorithms also factor the "age" of an area into the selection process, so the thrashing will normally be limited to two restarts. However, most operating systems will dispatch a new task while waiting for the disk subsystem to fill the requested area, and therefore the newly filled area will be "aged" somewhat by the time it finally gets used. Instruction continuation, utilized by the MC68010, MC68012, and MC68020, completely alleviates this problem because once an area has been successfully accessed, it is never needed again by that instruction.

Due to the virtual requirements and the need to accomplish this virtuality without massive operating system code, the MC68010, MC68012, and MC68020 have larger stack frames

IF YOU STILL USE PAPER FORMS TRY FORMMANAGER II!



Name	John Smith
Address	111 Board Way
City	San Francisco
State	CA
Zip	94111
Date	07-15-86

Prod No	Description	Price	QTY	Total
122-333-33	Gadget	10	100	1000.00
233-33-33	What is that	1	40	40.00
				1040.00

Data file: C:order.dat   F1 Help   Page 1 of 1   Record # 1 2

MEET FORMMANAGER II. It can help automate your paper office. More than a data manager, FormManager II lets you generate forms and organize them easily. You simply lay out your forms on the screen and fill in the data. FormManager II can calculate the tax due, total amount or even set up conditional calculations. More than 26 mathematical functions are included.

FormManager II can print on preprinted forms or just print on any plain paper. With a laser printer, you can even use different fonts on the same page.

HERE ARE SOME HIGHLIGHTS:
- Generate or use dBase III™ data files.
- Lay out forms on the screen with Forms Editor.
- Use the cursor keys to move from field to field.
- Fill in data easily with error checking.
- Print on pre-printed forms.
- Print on any plain paper with lines and boxes.
- Sort, search and modify any data records easily.
- Do complex calculation or even conditional statements.
- Report writer summarizes all your records.

SPECIAL LIMITED TIME OFFER ONLY **$99**

****

Attention, MS FORTRAN, PASCAL, C Programmers,

Forms Designer programming tool can save you time and effort producing professional screen forms. Don't struggle with the formatted I/O for screen data entry. Just use our Forms Editor to lay out the forms and call our library routines to do the job.

Comes with Forms Editor, runtime library, sample programs and library source code for PASCAL and C.

Only **$275.**

To Order Call
408-262-1054

dBase III is a trademark of Ashton-Tate, Inc.
FormManager and Forms Designer are trademarks of BIT Software, Inc.

bit **BIT Software, Inc.**
P.O. Box 360619
Milpitas, CA 95035

associated with the bus error exceptions (see figure 4). On the MC68000/008 processors, very little information must be saved for bus errors because the MPUs accommodate system faults rather than virtual faults. The MC68010/012/020 processors, on the other hand, save more internal information in support of instruction continuation. These variances in stack frames are buried in the exception handlers operating in the supervisory mode of an operating system. Only those exception handlers that deal with the maintenance of the supervisory stack or expect values at specific offsets on the supervisory stack are affected by the stack differences.

Thus, any exception handler code that is written to execute on one processor will require only minor or no changes to execute on another family member processor. These software changes are a normal part of upgrading a system, since memory management systems are typically upgraded from one system to the next—particularly when going from a nonvirtual memory environment to a virtual memory environment. However, the specific type of memory management implemented has no bearing on the compatibility, since the processor

The instruction continuation method protects the user from processor thrashing.

itself is not affected by the memory management scheme chosen.

PHYSICAL BUS INTERFACE

All members of the MC68000 family use the same simple asynchronous

(continued)

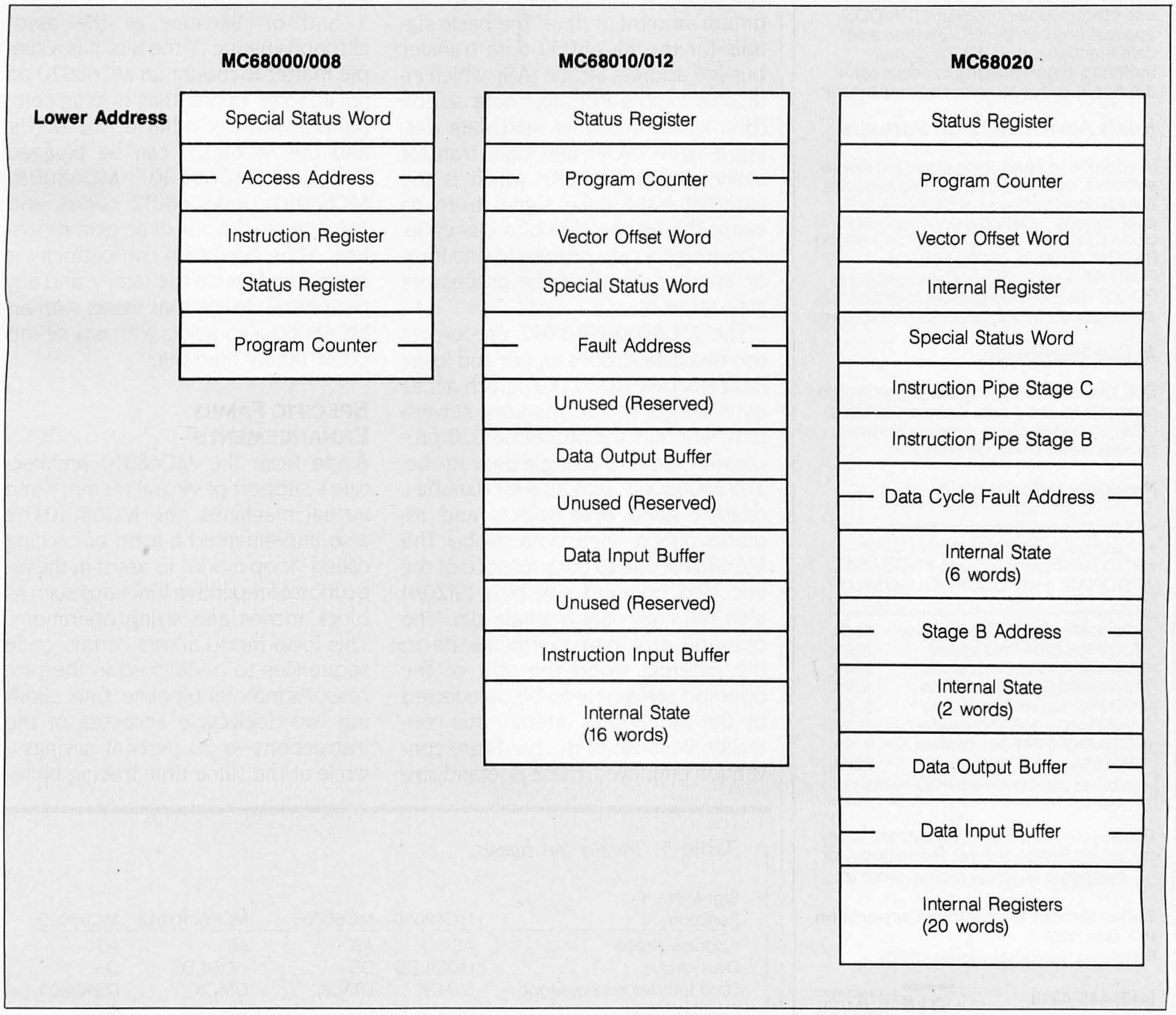


Figure 4: *Bus error stack frames.*

The MC68010/012 implemented the loop mode and support for program breakpoints.

bus interface to the outside world. On an asynchronous bus, the processor starts the bus cycle, and the external world controls the length of this cycle. Because of this, each portion of the memory subsystem (including memory-mapped peripherals) can respond to the bus cycle in its own optimum amount of time. The basic signals for the MC68000 data transfer bus are address strobe (AS), which indicates valid addresses; data strobe (DS), which indicates valid data during a write cycle; and data transfer acknowledge (DTACK), which is the external handshake signal used to cause the termination of a bus cycle. These signals are present in one form or another on all of the processors (see table 5).

The MC68000/010/012 processors use two data strobes (upper and lower data strobes, UDS/LDS), which act as byte selects to the memory subsystem, whereas the MC68008/020 processors use only a single data strobe. The MC68008, with its 8-bit data bus, doesn't need byte selects and requires only a single data strobe. The MC68020, due to the presence of the encoded operand size pins (SIZ0/1) also requires only a single pin. The operand size pins communicate to the external world the size of the operand remaining to be transferred by the MC68020's internal bus controller. Because of the hardware convention employed, these operand size pins may be combined with the low-order two address bits to generate the needed byte selects. The MC68020 dynamic bus sizing feature, which is enabled by encoding the data transfer acknowledge signals (DSACK0/1), allows the memory port size to change from 8 to 16 to 32 bits on a bus-cycle-by-bus-cycle basis (with one undefined-reserved encoding). Also, the MC68020 supports operand misalignment to the byte level.

The major difference in the external bus cycles is that the MC68000/008/010/012 execute a minimum bus cycle in four clock cycles, while the MC68020 executes its minimum cycle in only three clock cycles (see figures 5 and 6). Because of the asynchronous nature of the bus, it is a simple matter to mount an MC68020 on an adapter board that is plug-compatible with any other of the MPUs, and the MC68020 can be plugged into an MC68000, MC68008, MC68010, or MC68012 socket and can operate the bus of an existing system. Thus, hardware compatibility is maintained within the family, and any peripheral device that works with an MC68000 also works with any of the other family members.

SPECIFIC FAMILY ENHANCEMENTS

Aside from the MC68010 architecture's support of virtual memory and virtual machines, the MC68010/012 also implemented a form of caching called "loop mode" to assist in the execution of repetitive functions such as block moves and string operations. This loop mode allows certain code sequences to be latched in the processor's internal pipeline, thus allowing two-clock-cycle accesses of the instructions—a 50 percent savings—while at the same time freeing band-

Table 5: *Transfer bus signals.*

Signal Name Function	MC68000	MC68008	MC68010/012	MC68020
Address strobe	AS	AS	AS	AS
Data strobe	UDS/LDS	DS	UDS/LDS	DS
Data transfer acknowledge	DTACK	DTACK	DTACK	DSACK0/1
Operand size	n/a	n/a	n/a	SIZ0/1

Inquiry 121

68000 Reasons to Upgrade Your Apple II ...

Transform your Apple II, II+ or Apple //e into a powerful 32-bit supermicro. Simply plug in a PDQ 68000 coprocessor and PDQ highspeed RAM card; ProDOS and DOS acceleration and utility software is included.

the PDQ Super-32 System

The Highest Performance Apple II 68000 Coprocessor

Boosts Apple II Performance

PDQ's Applesoft acceleration is compatible with most off-the-shelf ProDOS and DOS Basic software, such as BPI General Accounting. PDQ's lightning-fast disk emulation results in dramatic performance improvements with Appleworks and other popular programs. ProDOS applications can use PDQ's time and date functions. And PDQ's printer buffering allows printing to occur while the Apple is free to perform another task.

Hosts Advanced 32-bit Software

In addition to enhancing standard Apple software, the PDQ System also provides access to a multitude of 68000-specific applications. Optional applications include CP/M 68K with C, the UCSD 68000 PASCAL System, 68000 FORTH, FORTRAN and LISP, and many others. PDQ'S 68000 Macroassembler/Editor is an integrated 68000 development system.

In Use Worldwide

ETC has been shipping advanced 68000 products for the Apple II since 1983. No other company offers a more complete, proven family of 68000 products.

Products with Integrity

- 15 day money back guarantee
- 5 year parts warranty, 1 year labor
- PDQ RAM cards include PDQBASIC, PDQRAM and PDQSPOOL software

PDQ 10Mhz 68000 Coprocessor	**$449.**
PDQ 256K RAM Card option	**$169.**
PDQ 1MEG RAM Card option	**$399.**
PDQ 2MEG RAM Card option	**$729.**
PDQ Macroassembler/Editor *	**$ 99.**
PDQBASIC+ 68000 Compiler *	**$ 79.**
UCSD 68000 PASCAL System *	**$ 99.**
CP/M 68K with C language *	**$290.**

(*requires 256K minimum PDQ memory)

Order your PDQ Super-32 System today by calling Enhancement Technology, or send your check or money order to

Enhancement Technology Corporation
P.O. Box 1267
Pittsfield, Massachusetts 01201

(413) 445-4219

ETC ENHANCEMENT TECHNOLOGY CORPORATION

Advanced 68000 coprocessors

width on the external data transfer bus. The sequences are composed of a 2-byte instruction, such as MOVE or ADD, which is followed immediately by a decrement and branch on condition (DBcc) instruction back to the 2-byte instruction. When the processor recognizes this sequence, it simply recycles the instructions and performs only data accesses to the outside world until the loop is terminated.

In addition to the loop mode, the MC68010/012 implemented special support for program breakpoints. Here, the recognition of a breakpoint *bit pattern* (one of eight specific illegal instruction encodings) causes the processor to execute a specific CPU space bus cycle during the exception processing for the illegal bit pattern to indicate that a breakpoint has been reached.

The MC68020 takes both of these features (loop mode and breakpoints) one step farther. The MC68020 has an on-chip 256-byte instruction cache, which increases overall processor performance from 40 percent to 80 percent, depending on the locality of reference exhibited by the code being executed. Code showing the greatest locality of reference (e.g., short loops) will show the greatest performance improvement by turning on the cache, while code that exhibits less (or no) locality of reference (long loops or in-line code) will show less (or no) performance increase. The cache is implemented on the MC68020 as a direct-mapped 64-entry by 32-bit cache. The supervisory programmer has control over this new resource by means of the cache control register.

The MC68020 enhancements to the MC68010 breakpoint function not only cause the processor to run the breakpoint acknowledge bus cycle just as before but also allow external hardware to supply a 16-bit instruction op code to be executed in place of taking the illegal instruction trap. This means that the MPU can execute the breakpoint a fixed number of times, substituting the replacement op code each time through the loop until the count expires and the breakpoint halts the loop.

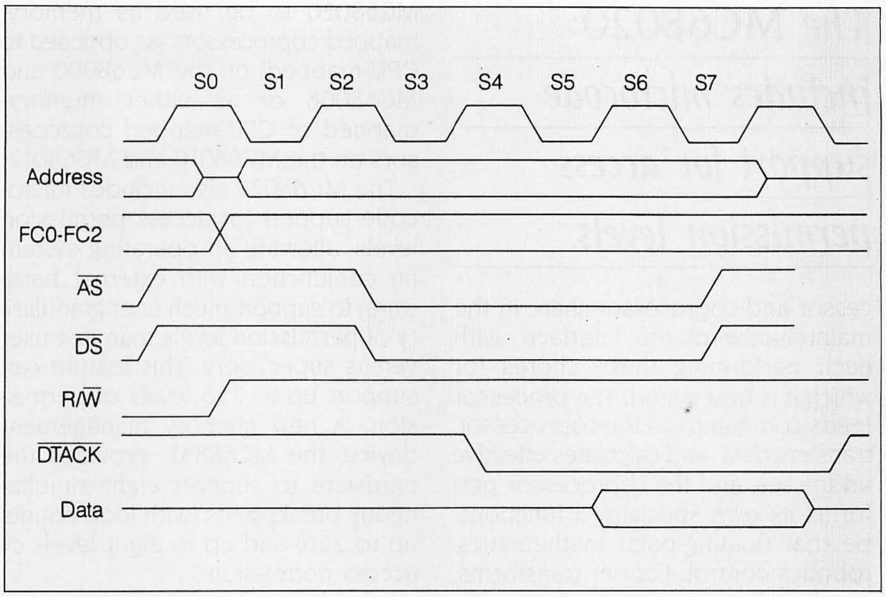


Figure 5: *The* READ *bus cycle for the* MC68000, MC68008, MC68010, *and* MC68012.

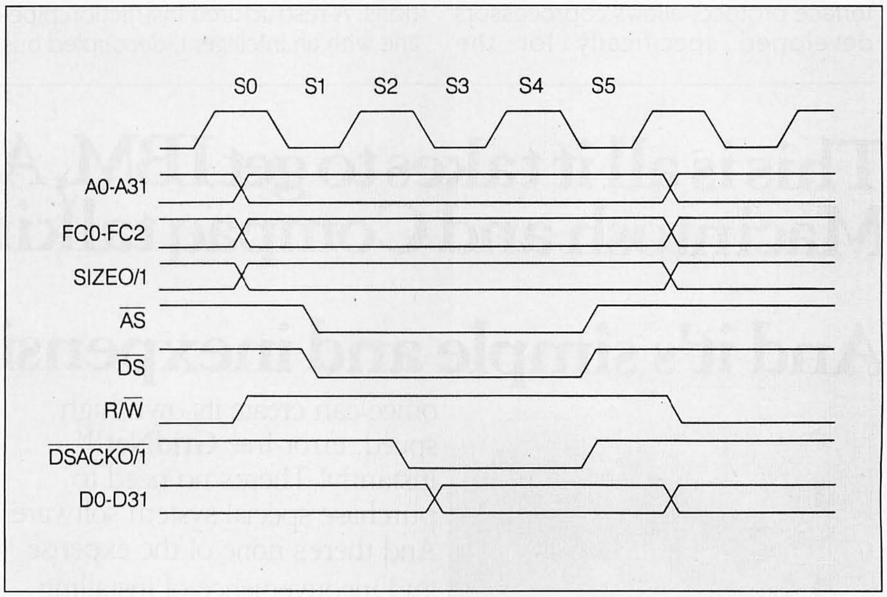


Figure 6: *The* READ *bus cycle for the* MC68020.

The MC68020 is the first processor in the family to offer inherent support for coprocessors. This coprocessor interface allows any coprocessor developed for the MC68020 also to be used on any other of the family members. This is accomplished by making the maintenance of the coprocessor interface reliant on standard read/write bus cycles. Because the interface is generic in nature, new processor designs will not require reworking existing coprocessors, and new coprocessor designs will not require reworking existing processors. In fact, the coprocessor interface gives users of the microprocessor an opportunity to develop their own custom coprocessors.

The extensibility offered by the coprocessor interface is one of the most powerful features of the MC68000 family architecture. Both the pro-

(continued)

The MC68020 includes microcode support for access permission levels.

cessor and coprocessor share in the maintenance of the interface, with each performing those chores for which it is best suited: The processor feeds commands to the coprocessor, transfers data, and calculates effective addresses, and the coprocessor performs its own specialized functions, be that floating-point mathematics, robotics control, Fourier transforms, graphics functions, or anything else. While the MC68020 offers implicit microcode support for the coprocessor interface, emulation of the interface protocol allows coprocessors developed specifically for the MC68020 to be used as memory-mapped coprocessors (as opposed to CPU-mapped) on the MC68000 and MC68008 or as either memory-mapped or CPU-mapped coprocessors on the MC68010 and MC68012.

The MC68020 also includes microcode support for access permission levels, allowing an operating system (in conjunction with external hardware) to support much finer granularity of permission levels than just user versus supervisory. This feature can support up to 256 levels of permission. A new memory management device, the MC68851, provides the hardware to support eight simultaneous breakpoints (with loop counts up to 256) and up to eight levels of access permission.

In addition, the MC68020 has full 32-bit ALUs and a 32-bit barrel shifter for improved performance on arithmetic, logical, and bit-field instructions. A restructured instruction pipeline with an intelligent, decoupled bus controller allows a high degree of concurrency in instruction execution with external bus activity.

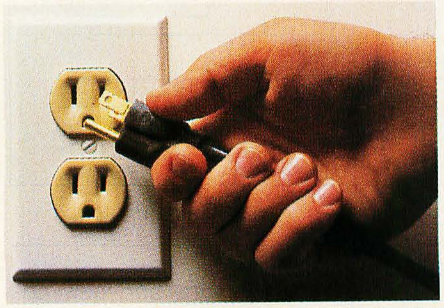
CONCLUSIONS

In the final analysis, the name of the game is compatibility. And the bottom line of compatibility between members of the MC68000 family shows that, for user code, all family members are 100 percent upwardly compatible for object code, while supervisory-level code only requires changes in supervisory-level exception handlers that use specific information on the supervisory stacks. Hardware compatibility is also good. In fact, an MC68010 can be directly inserted into an MC68000 socket, and an MC68020 could also be used with the appropriate adapter board.

The MC68000 family provides not only high performance in a commercial microprocessor, but also compatibility, protecting the user's software investment. ■

This is all it takes to get IBM, Apple, Tandy, Macintosh and Compaq talking to each other.

And it's simple and inexpensive.



Now a powerful **GridNet**™ Local Computer Communications Network is as close as your nearest electrical outlet.

Thanks to GridComm's Communicators, virtually any office can create its own high speed, error-free **GridNet**™... instantly! There's no need to purchase special system software. And there's none of the expense and inconvenience of installing additional wiring. Because a **GridNet**™ utilizes your existing electrical grid as a highly-effective data transmission medium.

And a **GridNet**™ will support IBM PC, XT, AT, and all IBM compatibles, Apple and Macintosh, Hewlett-Packard, Tandy, AT&T, Epson, NEC...all the most popular micros and peripherals, in the same network. Freely sharing data, with complete connectivity and effective compatibility.

The cost? As much as 60% below that of specially-wired systems.

Find out just how easy it is to increase your office's productivity. See your local dealer, or contact "The Great Communicator" for more information.

GRIDCOMM
THE GREAT COMMUNICATOR

For information call 203-790-9077. P.O. Box 2779, Danbury, Connecticut 06813-2779

Trademarks: IBM, PC, XT, AT/International Business Machines Corp.—Apple/Apple Computer Inc.— Macintosh/McIntosh Laboratory Inc. —Tandy/Tandy Corp.—Compaq/Compaq Computer Corp. Hewlett-Packard/Hewlett-Packard Co. —Epson/Epson America Inc.—AT&T/American Telephone and Telegraph Co.—NEC/NEC Home Electronics (U.S.A.) Inc.

Inquiry 148